```sql
USE mvc_team_four;

## Question 1
/*Creates a view that displays the amount of collisions for each
unique contributing factor sorted by most to least collisions.*/
DROP VIEW IF EXISTS leading_causes;
CREATE VIEW leading_causes AS
SELECT
      cf.CONTRIBUTING_FACTOR AS Factors,
    COUNT(*) AS Collisions
FROM contributing_factor cf
JOIN vehicle_contributing_factor vcf ON cf.CONTRIBUTING_FACTOR_ID =
vcf.CONTRIBUTING_FACTOR_ID
GROUP BY cf.CONTRIBUTING_FACTOR
ORDER BY Collisions DESC;
SELECT * FROM leading_causes;


## Question 2
/*Creates a view that displays the amount of collisions for
each unique vehicle make and its model sorted by most to least
collisions.*/
DROP VIEW IF EXISTS make_model_collisions;
CREATE VIEW make_model_collisions AS
SELECT distinct
      vi.VEHICLE_MAKE AS VehicleMake,
    vi.VEHICLE_TYPE AS VehicleModel,
    COUNT(vc.COLLISION_ID) AS Collisions
FROM vehicle_information vi
JOIN vehicle_collisions vc ON vi.UNIQUE_ID = vc.UNIQUE_ID
GROUP BY vi.VEHICLE_MAKE, vi.VEHICLE_TYPE
ORDER BY Collisions DESC;
SELECT * FROM make_model_collisions;


## Question 3
/*Creates a view that displays the amount of collisions for
every two hour intervals of the day sorted by most to least collisions.*/
DROP VIEW IF EXISTS collision_time_frame;
CREATE VIEW collision_time_frame AS
SELECT
      CONCAT(
                  FLOOR(HOUR(ci.CRASH_TIME) / 2) * 2, ':00 - ',
                  FLOOR(HOUR(ci.CRASH_TIME) / 2) * 2 + 1, ':59'
      ) AS Time,
    COUNT(vc.COLLISION_ID) AS Collisions
FROM collision_information ci
JOIN vehicle_collisions vc ON ci.COLLISION_ID = vc.COLLISION_ID
GROUP BY Time
ORDER BY Collisions DESC;
SELECT * FROM collision_time_frame;


/*Question 4
```

```
includes: JOIN, FILTER, AGGREGATE, LINKING, SUB-QUERY
 Creates a view that displays the amount of collisions for every season,
ignoring the year.*/
DROP VIEW IF EXISTS collisions_in_seasons;
CREATE VIEW collisions_in_seasons AS
SELECT COUNT(CRASH_DATE) AS winter_collisions,
-- subquery for spring
(SELECT COUNT(CRASH_DATE)
FROM vehicle_information
JOIN vehicle_collisions
USING(UNIQUE_ID)
JOIN collision_information
USING(COLLISION_ID)
WHERE MONTH(CRASH_DATE) BETWEEN 3 AND 5) AS spring_collisions,
-- subquery for summer
(SELECT COUNT(CRASH_DATE)
FROM vehicle_information
JOIN vehicle_collisions
USING(UNIQUE_ID)
JOIN collision_information
USING(COLLISION_ID)
WHERE MONTH(CRASH_DATE) BETWEEN 6 AND 8) AS summer_collisions,
-- subquery for fall
(SELECT COUNT(CRASH_DATE)
FROM vehicle_information
JOIN vehicle_collisions
USING(UNIQUE_ID)
JOIN collision_information
USING(COLLISION_ID)
WHERE MONTH(CRASH_DATE) BETWEEN 9 AND 11) AS fall_collisions
FROM vehicle_information
JOIN vehicle_collisions
USING(UNIQUE_ID)
JOIN collision_information
USING(COLLISION_ID)
WHERE MONTH(CRASH_DATE) = 12 OR MONTH(CRASH_DATE) BETWEEN 1 AND 2;

SELECT * FROM collisions_in_seasons;


/*Question 5,
made a procedure since older vs newer model is super dependent on what
year the person asking the question is in*/
-- includes: FILTER, AGGREGATE, SUB-QUERY
DROP PROCEDURE IF EXISTS older_vs_newer;
DELIMITER //
CREATE PROCEDURE older_vs_newer (
      year_param       VARCHAR(4)
      )
BEGIN
      SELECT COUNT(VEHICLE_YEAR) AS num_older, (SELECT COUNT(VEHICLE_YEAR)
FROM vehicle_information WHERE VEHICLE_YEAR >= year_param) AS num_newer
      FROM vehicle_information
      WHERE VEHICLE_YEAR < year_param;
```

```
END //
DELIMITER ;
/*just for testing*/
CALL older_vs_newer('2010');


-- Question 6
/*Creates a view to calculate the average number of occupants in vehicles
involved in collisions.*/
USE mvc_team_four;
DROP VIEW IF EXISTS average_occupants_in_collisions;
CREATE VIEW average_occupants_in_collisions AS
SELECT AVG(VEHICLE_OCCUPANTS) AS AVG_OCCUPANTS
FROM vehicle_information;

SELECT * FROM average_occupants_in_collisions;


/*Question 7
Creates a view that displays the number of collisions caused by male
drivers compared to female drivers.*/
-- includes: JOIN, FILTER, AGGREGATE, LINKING, SUB-QUERY
DROP VIEW IF EXISTS driver_gender_num;
CREATE VIEW driver_gender_num AS
SELECT COUNT(DRIVER_SEX) AS num_men,
-- subquery for women
(SELECT COUNT(DRIVER_SEX) FROM mvc_team_four.vehicle_information JOIN
vehicle_drivers
USING(UNIQUE_ID)
JOIN driver_information
USING(DRIVER_ID)
WHERE DRIVER_SEX = 'F') AS num_women
FROM mvc_team_four.vehicle_information
JOIN vehicle_drivers
USING(UNIQUE_ID)
JOIN driver_information
USING(DRIVER_ID)
WHERE DRIVER_SEX = 'M';

SELECT * FROM driver_gender_num;


-- Question 8
/*Creates a view that displays whether the license status of the driver of
the collisions
and the amount of collisions for each status*/
USE mvc_team_four;
DROP VIEW IF EXISTS license_registration_collisions;
CREATE VIEW license_registration_collisions AS
SELECT
    dlj.DRIVER_LICENSE_STATUS AS "License Status",
    COUNT(*) AS Collisions
FROM driver_license_jurisdiction dlj
```

```
JOIN driver_information di ON dlj.DRIVER_LICENSE_JURISDICTION_ID =
di.DRIVER_LICENSE_JURISDICTION_ID
GROUP BY dlj.DRIVER_LICENSE_STATUS
ORDER BY Collisions DESC;

SELECT * FROM license_registration_collisions;


-- Question 9
/*Creates a view that displays how the vehicle was moving when the crash
occured and the
amount of collisions associated with the movement. Sorted by amount of
collisions from most to least.*/
USE mvc_team_four;
DROP VIEW IF EXISTS pre_crash_information;
CREATE VIEW pre_crash_information AS
SELECT
    vi.PRE_CRASH AS "Vehicle Status",
    COUNT(*) AS Collisions
FROM vehicle_information vi
GROUP BY vi.PRE_CRASH
ORDER BY Collisions DESC;
SELECT * FROM pre_crash_information;
```